

OTOMATISASI PEMODELAN COAD YOURDAN

Febri Nova Lenti

STMIK AKAKOM Yogyakarta
Jl. Raya Janti 143, Karangjambe Yogyakarta
email: febri@akakom.ac.id

ABSTRAK

Paradigma analisis dan desain berorientasi objek memiliki banyak varian metode. salah satu variannya adalah metode *Coad-Yourdan*. Metode *Coad-Yourdan* adalah metode yang berpusat pada data dan relasi. Jika analisis dan desain berorientasi objek dilakukan melalui metode *Coad-Yourdan* kemudian divisualisasikan dengan menggunakan tool bahasa pemodelan UML seperti; *Rational Rose* atau *Argo UML* maka dibutuhkan beberapa konversi. Oleh sebab itu untuk memudahkan pengguna metode *Coad yourdan* memvisualisasikan hasil pemodelannya maka dikembangkan suatu aplikasi yang akan mengotomatisasi pemodelan *Coad Yourdan*. Aplikasi ini di sebut *Coad Yourdan Modelling Language (CYMoL)*. CYMoL akan memvisualisasikan diagram things; object, class dan subject, Juga diagram struktur berupa; whole part dan genspec serta dilengkapi dengan fasilitas menurunkan skema program dalam bahasa Java.

Keywords :metode *Coad Yourdan*, object, class, subject, whole part, Genspec,

1. PENDAHULUAN

Saat ini penggunaan analisis dan desain berorientasi objek dalam menyelesaikan permasalahan di beberapa bidang aplikasi sangat populer. Hal ini dikarenakan beberapa keuntungan dan kelebihan yang ada pada analisis dan desain berorientasi objek dibandingkan dengan paradigma SSAD (*Structured System Analysis and Design*). Karena analisis dan desain berorientasi objek ini memiliki banyak varian metode dimana tiap metode memiliki notasi sendiri-sendiri maka OMG (*Object Management Group*) yang merupakan sebuah konsorsium dibidang teknik berorientasi objek mengeluarkan *Request for Proposal (RFP)* untuk mendorong perusahaan-perusahaan sistem informasi, developer software, dan para user sistem komputer membuat sebuah RFP bersama sebagai respon RFP dari OMG untuk bahasa pemodelan UML (*Unified Modelling Language*) yang merupakan sebuah bahasa pemodelan yang bersifat universal untuk semua metode-metode yang ada dalam paradigma analisis dan desain berorientasi objek. Sebuah perusahaan yaitu *Rational Software* telah membentuk konsorsium untuk meresmikan UML untuk digunakan sebagai bahasa standar dalam analisis dan desain berorientasi objek.

UML ini kemudian di otomatisasi dengan dikembangkannya beberapa perangkat lunak Bantu seperti *Rational Rose*, *Argo UML*, dan lain lain. UML merupakan bahasa universal sehingga tidak dapat memvisualisasikan secara langsung notasi-

notasi yang dipakai pada beberapa metode-metode yang ada dalam paradigma analisis dan desain berorientasi objek. Untuk itu diperlukan beberapa konversi jika hasil analisis dan desain menurut metode tertentu ingin divisualisasikan menggunakan tool UML terutama metode *Coad Yourdan*. Metode *Coad-Yourdan* adalah metode visualisasi yang berpusat pada data dan relasi. Pada tulisan ini akan dibahas mengenai prototype bahasa pemodelan untuk metode *Coad Yourdan*.

2. LANDASAN TEORI

2.1 Konsep dan Konstruksi Coad Yourdan

Objek adalah: abstraksi dari sebuah entitas nyata/tidak nyata (*tangible intangible*) yang informasinya harus diingat/disimpan: nilai-nilai atribut dan layanan-layanan eksklusif dienkapsulasi

Kelas adalah :

deskripsi dari satu atau lebih objek dengan sejumlah atribut dan layanan yang seragam termasuk deskripsi tentang cara membuat objek dari kelas tersebut

Atribut adalah :

sejumlah data (informasi keadaan) di mana tiap objek dari suatu kelas mempunyai nilai tersendiri

Class-&-Object adalah :

suatu istilah yang berarti "suatu kelas dan objek-objek yang ada pada kelas tersebut "

Subjek adalah :

Mekanisme untuk membagi model yang besar dan kompleks. Subjek juga berguna untuk mengatur paket kerja pada proyek-proyek besar berdasarkan hasil penyelidikan awal dengan OOA.

Layanan (Service) adalah :

kelakuan spesifik yang dilakukan oleh objek yang menjadi tanggung jawab objek tersebut

Keadaan/Status (State):

Status dari sebuah objek adalah gabungan dari nilai-nilai atribut objek tersebut

Transisi (Transition): adalah perubahan status

Kondisi (Condition): aksi If-precondition-trigger-terminate

Text Block: teks

Loop: aksi While-Do-Repeat-trigger-terminate

2.2 Hubungan antar Objek

Pada metode Coad Yourdan terdapat beberapa struktur hubungan antar obyek yaitu:

2.2.1 Struktur Whole-Part

dimana satu objek (yang mewakili Whole) dapat didekomposisi menjadi objek-objek lain . Terdapat 3 variasi struktur whole-part:

1. Assembly-Parts : yaitu satu obyek berperan sebagai assembly dan obyek obyek lainnya merupakan bagian bagian yang merakitnya . contoh: sebuah mobil mempunyai roda, mesin, chasis
2. Container-Contents : yaitu satu obyek berperan sebagai wadah / container dan obyek lainnya sebagai elemen / isi dari container. contoh: sebuah kotak berisi sejumlah paku, benang, boneka
3. Collection-Members : yaitu satu obyek berperan sebagai koleksi / organisasi dan obyek lainnya menjadi anggotanya. contoh: sebuah organisasi mempunyai sejumlah analis, manajer

Hubungan Whole-Part relationships dapat memiliki rentang spesifik, seperti layaknya konsep kardinalitas pada pemodelan E-R Instance connections (Hubungan asosiasi) Yaitu koneksi yang diperlukan oleh sebuah objek dengan objek lain dalam rangka memenuhi tanggung jawabnya.

Contoh: Sebuah objek Orang bekerja pada objek Kantor Juga bisa memiliki rentang

2.2.2 Message Connections

Memodelkan ketergantungan pemrosesan sebuah objek yang dinyatakan dengan kebutuhan atas layanan-layanan dari objek lain dalam rangka

memenuhi tanggung jawabnya (layanan yang disediakan olehnya) sendiri.

Contoh:

Untuk memenuhi tanggung jawab (layanan) "permohonan cuti", objek Orang perlu layanan "persetujuan cuti" dari objek Manajer

2.3 Hubungan antar-Kelas

Struktur Generalization-Specialization / Gen-Spec (Hubungan pewarisan) Mendefinisikan hirarki pewarisan untuk kelas-kelas yang merupakan spesialisasi dari kelas lain yang lebih umum (general). Sebuah kelas bisa mewarisi sifat dari sebuah superclass (kelas general) yang disebut dengan pewarisan tunggal (single inheritance) atau dari sejumlah superclass yang disebut dengan pewarisan ganda (multiple inheritance).

Contoh: seorang Manajer adalah spesialisasi dari Orang.

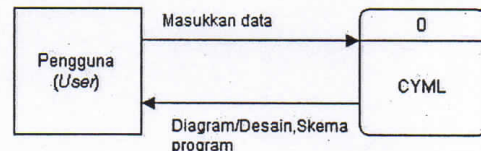
3. PEMBAHASAN

3.1 Rancangan CYMoL

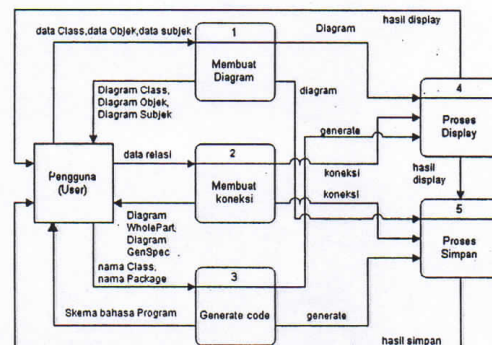
CYMoL mempunyai 3 utilitas yaitu:

1. Membuat dan menampilkan things; object,class, subjek
2. Membuat dan menampilkan struktur; whole part dan genspec
3. Menampilkan skema program dalam bahasa Java

Diagram aliran data (DAD) pada aplikasi ini ditunjukkan dalam diagram konteks (gambar 1) dan DAD Level 1 (Gambar 2)

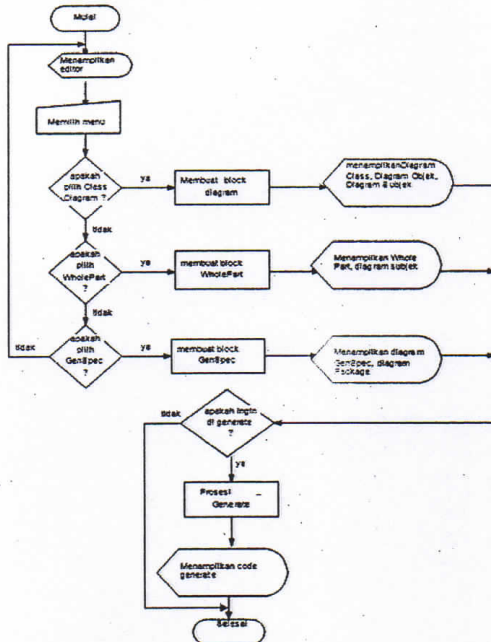


Gambar 1. Diagram Konteks CYMoL



Gambar 2. DAD Level 1 CYMoL

CYMoL ini memiliki mekanisme kerja yang digambarkan pada *flowchart* dalam gambar 3. Dari gambar 3 dapat dilihat ada 4 proses utama, yaitu: menampilkan block diagram, menampilkan block Whole Part, menampilkan block GenSpec dan Proses Generate



Gambar 3. Mekanisme kerja CYMoL

Kemudian aplikasi CYMol ini memiliki beberapa fungsi / prosedur sebagai berikut :

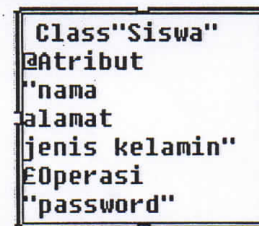
1. **Create Diagram**
{untuk mebuat diagram}
2. **Input relasi.**
{untuk memasukkan relasi / koneksi ke diagram}
3. **Button l**
{untuk menggambar subject pada editor dengan cara memanggil *classModule cBlock*.}
4. **Button Class**
{untuk menggambar diagram class pada editor dengan sebelumnya memanggil *classModule cBlock*}
5. **Button WholePart**
{untuk menggambar notasi Whole Part pada editor dengan sebelumnya memanggil *classModule cBlock*}
6. **Button GenSpec**
{untuk menggambar notasi GenSpec pada editor dengan sebelumnya memanggil *classModule cBlock*}
7. **Button Koneksi WholePart**
{untuk membuat koneksi Whole Part berupa garis lurus dengan sebelumnya memanggil *Class Module cLine*.}

8. **Button Koneksi GenSpec**
{untuk membuat koneksi GenSpec berupa garis lurus putus-putus}
9. **Button DeleteBlock**
{untuk menghapus block dengan sebelumnya memanggil *Class Module Block_Collection*}
10. **Button DeleteLine**
{untuk menghapus garis Koneksi dengan sebelumnya memanggil *Class Module Line_Collection*}
11. **Button Edit Class**
{untuk mengedit *Block Class Diagram*, pengeditan pada *Block* ini meliputi : pemberian nama *Class*, pengisian atribut, dan pengisian operasi. Gambar Button Edit Class Diagram }
12. **Button Edit Package**

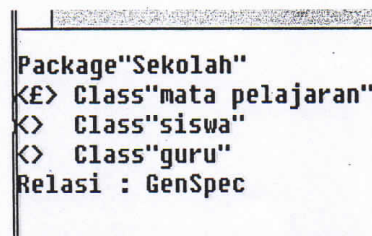
{untuk pengeditan *block Package*, pengeditan yang dilakukan meliputi: pemberian nama *Package*, penginputan nama-nama *Class Diagram* yang akan di *Package*, dan menampilkan pemberitahuan relasi yang digunakan pada *Package*. Relasi yang digunakan pada *Package* selalu bernilai "Relasi: GenSpec". Ini dikarenakan *Package* yang digunakan hanya untuk relasi *GenSpec* saja.}

3.2 PENGUJIAN

Untuk memulai membuat sebuah diagram klik pada menu Diagram lalu *CreateDiagram*. Disini dapat dibuat sebuah diagram obyek/class dan diagram package. Gambar 4 dan gambar 5 menampilkan diagram obyek dan package yang telah dibuat. Pada aplikasi atau software ini terdapat dua struktur yaitu *WholePart* dan *GenSpec*. Untuk menjalankan mode tersebut harus memilih salah satu mode pemodelan, dan akan ditunjukkan di status bar.



Gambar 4. Class Diagram



Gambar 5 Package

Diagram class maupun package ini dapat digenerate ke dalam program Java dengan skema program seperti ditunjukkan pada gambar 6 dan gambar 7.

```

JavaLanguage
public class Siswa
{
    //tipe code disini dibawah adalah contoh kode bisa dihapus dan diganti dengan source code kamu
    sendir//
    private int x;
    public Siswa()
    {
        x=0;
    }
    // contoh method - ganti code dengan milikmu sendiri
    @param y contoh parameter dari method
    @return sum dari x dan y//
    public int sampleMethod(int y)
    {
        return x + y;
    }
}
Contoh Pemrograman java "Have Nice Dream" 11:54 PM 1/27/2011

```

Gambar 6 Skema program Java Class diagram

Skema Program Java untuk Package

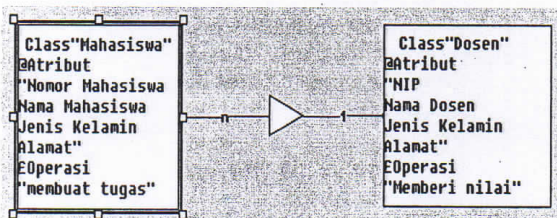
```

JavaLanguage
package Sekolah;
public class Mata Pelajaran
{
    sendir// //tipe code atau method disini diisi dengan source code kamu
    public Mata Pelajaran()
    {
    }
    // Main Method
    public static void main(String args[])
    {
    }
}
=====
public class Siswa extends Mata Pelajaran
{
    sendir// //tipe code atau method disini diisi dengan source code kamu
    public Siswa()
    {
    }
    // Main Method
    public static void main(String args[])
    {
    }
}
Contoh Pemrograman java "Have Nice Dream"

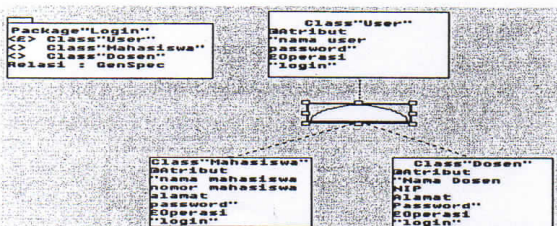
```

Gambar 7 Skema program Java untuk Package

Pada prototype ini juga dapat ditampilkan struktur Whole Part dan Genspec, dengan notasi Coad Yourdan seperti yang ditunjukkan oleh gambar 3.8 dan gambar 3.9



Gambar 8 Contoh Mode WholePart



Gambar 9 Contoh hasil struktur Genspec

Pada gambar diatas terlihat bahwa Class User merupakan SuperClass sedangkan Class Mahasiswa dan Class Dosen sebagai SubClass. Karena ada

karakteristik yang diwariskan yaitu *password*. Dan pada SubClass(dalam hal ini Class Mahasiswa dan Class Dosen),memiliki detail atau karakteristik sendiri(kelengkapan pada atribut) yang tidak dimiliki oleh SuperClass.

Selanjutnya untuk hasil generate codenya ditunjukkan oleh gambar 10

```

JavaLanguage
package Login;
public class User
{
    //tipe code atau method disini diisi dengan source code
    kamu sendiri//
    public User()
    {
    }
    // Main Method
    public static void main(String args[])
    {
    }
}
=====
public class Mahasiswa extends User
{
    //tipe code atau method disini diisi dengan source code
    kamu sendiri//
    public Mahasiswa()
    {
    }
    // Main Method
    public static void main(String args[])
    {
    }
}
Contoh Pemrograman java "Have Nice Dream"

```

Gambar 10 contoh hasil generate GenSpec

4. PENUTUP

Prototype bahasa pemodelan untuk Metode Coad Yourdan (CYMoL) ini dapat membantu pengguna metode tersebut untuk melakukan visualisasi pemodelan yang dilakukannya dengan menggunakan metode Coad Yourdan dengan mudah tanpa melakukan konversi lagi dan dapat segera dilanjutkan ke tahap implementasi karena kode skema program sudah langsung diturunkan. Untuk lebih meningkatkan manfaat CYMoL ini dapat ditambahkan hal hal sebagai berikut :

1. Sistem ini hanya menampilkan relasi *WholePart* dan *GenSpec* sehingga dalam pengembangan lebih lanjut dapat menambahkan beberapa relasi yang lain.
2. Generate Code bisa ditambahkan untuk beberapa bahasa pemrograman, tidak hanya Java

DAFTAR PUSTAKA

- [1] Hariyanto,Bambang, *Rekayasa Sistem Berorientasi Objek*, Informatika Bandung, Bandung, 2004.
- [2] Febri Nova Lenti,,*Penerapan OOA dan OOD Coad And Yourdan Untuk Analisis Dan Perancangan Sistem Informasi Wartel*, STMIK AKAKOM, Yogyakarta, 2002.
- [3] Febri Nova Lenti, *Forward Engineering Pada UML Dengan Menggunakan Rational Rose*

Studi Kasus Sistem Informasi Wartel, STMIK AKAKOM, Yogyakarta, 2006.

- [4] Sholiq, *Pemodelan Sistem Informasi Berorientasi Objek dengan UML*, Graha Ilmu, Yogyakarta, 2006.

- [5] Suhendar, A, dan Hariman Gunadi, *Visual Modelling Menggunakan UML dan Rational Rose*, Informatika Bandung, Bandung, 2002.